UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

ORACLE AMERICA, INC. Plaintiff, v. GOOGLE INC. Defendant.	Case No. CV 10-03561 WHA

OPENING EXPERT REPORT OF JOHN C. MITCHELL REGARDING PATENT INFRINGEMENT

SUBMITTED ON BEHALF OF PLAINTIFF ORACLE AMERICA, INC.

CONFIDENTIAL PURSUANT TO PROTECTIVE ORDER
Highly Confidential – Attorneys Eyes Only

pa-1460906

EXHIBIT 2-5

I, John C. Mitchell, Ph.D., submit the following expert report ("Opening Patent Infringement Report") on behalf of plaintiff Oracle America, Inc. ("Oracle"):

I. INTRODUCTION

1. I have detailed my retention, scope of work performed, expected testimony, compensation, and qualifications in my Opening Copyright Report.

2. I have also detailed a background on Java, Android, and Android's adoption of Java. I included a discussion of Google's alternatives to adopting Java and why each potential alternative would have failed to enable Google's own strategies in relation to Android.

3. Instead of repeating that content here, I incorporate my Opening Copyright Infringement Report by reference.

II. MATERIALS CONSIDERED AND RELIED ON

4. In arriving at my opinions in this case, I have considered a number of different sources of information which are identified in attached Exhibit B-Supplement and referenced in my report.

5. In particular, I have reviewed the patents-in-suit and their respective file histories; the Court's Claim Construction Order issued on May 9, 2011; and Android source code, videos, and documentation made publicly available and produced by Google during the course of discovery.

6. In support of my analysis and rendered opinions, I intend to rely on the performance benchmark and testing analysis completed by Robert ("Bob") Vandette, Noel Poore, and Erez Landau, as detailed in their respective summaries and reports submitted to Google with my Opening Patent Infringement Report. This work was conducted at my request and direction. I engaged in numerous conversations with these Java engineers in carrying out this work.

7. In addition to the materials specifically identified, I may provide further exhibits to be used as a summary of or support for my opinions.

B. The Claimed Inventions in the Patents-in-Suit are Necessary to Achieve Sufficient Performance and Security

38. Sun Microsystems (now Oracle) devised clever and innovative ways to achieve performance and security, captured in the seven asserted patents. These inventions cut across the Java platform – and likewise across the Android platform – addressing significant needs in each area. Google used these inventions directly to its advantage in the Android platform, at corresponding points in Google's parallel development and execution process. The benefits are significant and measureable, as discussed below in this report and in the performance and testing analyses reported by Robert ("Bob") Vandette, Noel Poore, and Erez Landau.

1. The Claimed Inventions Are Critical to Android Achieving Satisfactory Performance

39. ***'104 patent (Reference Resolution).*** In my opinion, the claimed invention of the '104 reissue patent is essential to the performance of Android's Dalvik virtual machine. In the absence of using the claimed invention, Android's Dalvik virtual machine would have to determine references on each encounter instead of resolving once and re-using the resolved information, thereby slowing down execution of the virtual machine.

40. I confirmed my qualitative analysis through directing performance benchmark work conducted by Oracle Java engineer Bob Vandette in consultation with Oracle Java engineer Dr. Peter B. Kessler. The source code modifications were based on a number of discussions I had with Oracle Java engineers Bob Vandette and Peter Kessler, my review of the '104 patent, my review of Oracle's infringement contentions submitted to Google on April 1, 2011, and my understanding of the Android source code that satisfies the claims of the '104 patent.

41. The performance benchmark testing results show an execution speed improvement of as much as 13 times. This means applications can execute as much as 13 times faster using the invention expressed in the '104 patent.

42. Therefore, it is my opinion that the performance of Android would be crippled without the benefit of using the '104 patent.

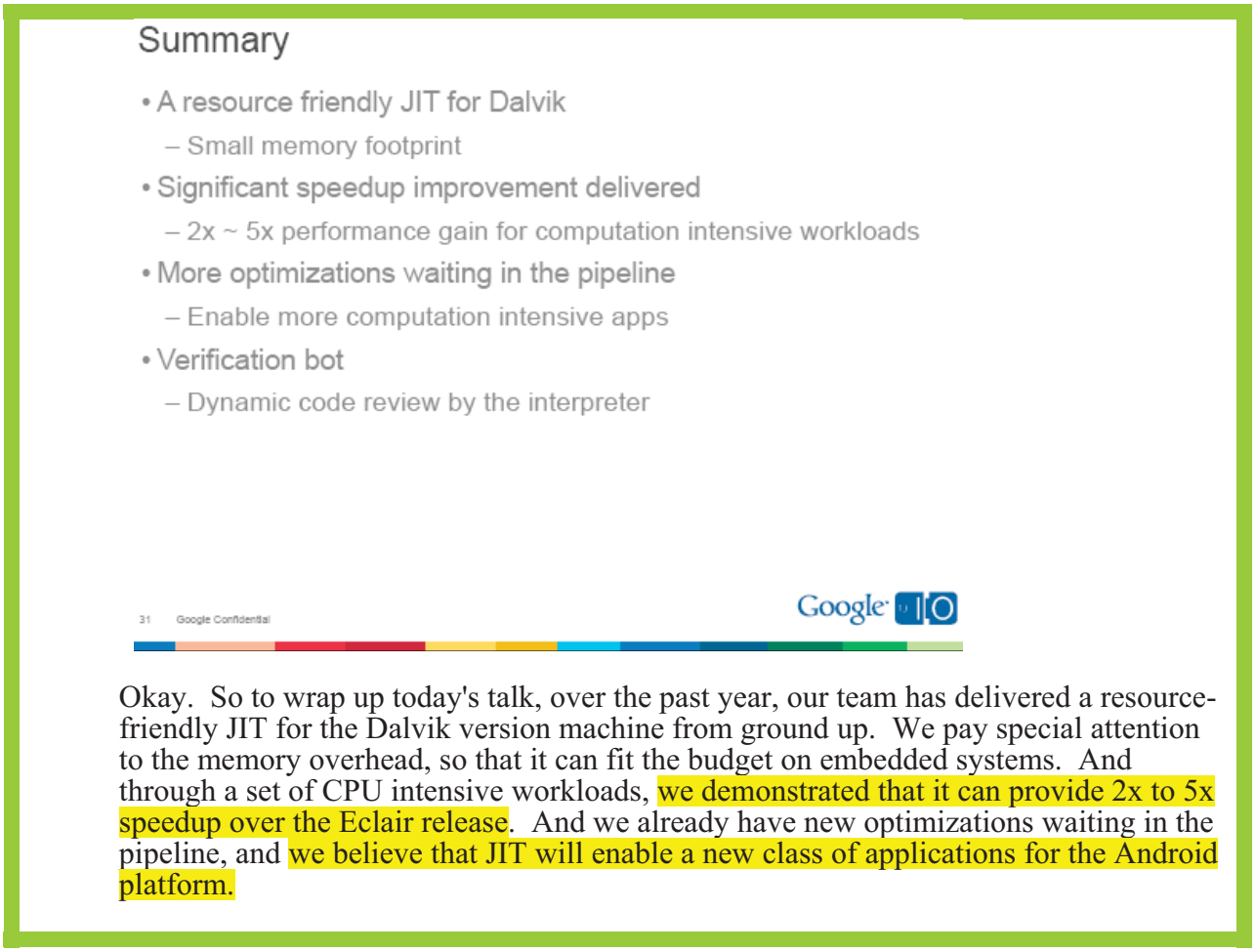
43. I intend to rely on the summary and report submitted by Bob Vandette, submitted to Google with my report, to support my opinion.

44. ***'205 patent (Hybrid Code Execution).*** The '205 patent involves generating and executing native instructions instead of virtual machine instructions. It is my opinion that Android infringes the '205 patent in two ways: (1) just-in-time compilation with an indication of whether there is native code for execution or not through a table entry and (2) inlining.

45. I note that Google made an affirmative choice to add just-in-time compilation features. (*See, e.g.*, Google I/O 2010 Video entitled "A JIT Compiler for Android's Dalvik VM," presented by Ben Cheng and Bill Buzbee (Google's Android Team), *available at* http://developer.android.com/videos/index.html#v=Ls0tM-c4Vfo (last visited Nov. 5, 2010)):

> "Now, interpreters, the way they work, the- they will go out and fetch one Dalvik instruction at a time – we call them Dalvik bytecodes – pull the instruction apart, see what it is, that's called the decode phase, and then go ahead and interpret it or execute it. And that execution is done by using actually the host instructions on the host processor. But what you have in effect there with interpretation is an extra stage of execution. So you have to pull up the bytecode, figure out what it is, then use host instructions to carry it out, and the CPU will pick up the host instructions and, and, and execute them. And that extra level of, of evaluation is what gives interpreters a bit of a bad name for being slow. I mean, there's some great reasons why you would want to use an interpreter, but the down side is that they're often a bit slower than native- natively compiled code. …. Now, putting a JIT on Android has been something we've talked about for a long time, internally, but the big question was what kind of a JIT can we fit into an Android system?...So our key, our key requirements going into this process: we needed to find a JIT system that could deliver performance using a very small amount of memory."

46. And the performance impact resulting from Google's choice to include JIT in Android is significant: Google admits (indeed, advertises) 2 times to 5 times faster execution as a result of JIT:



Okay. So to wrap up today's talk, over the past year, our team has delivered a resource-friendly JIT for the Dalvik version machine from ground up. We pay special attention to the memory overhead, so that it can fit the budget on embedded systems. And through a set of CPU intensive workloads, we demonstrated that it can provide 2x to 5x speedup over the Eclair release. And we already have new optimizations waiting in the pipeline, and we believe that JIT will enable a new class of applications for the Android platform.

(Google I/O 2010 Video entitled "A JIT Compiler for Android's Dalvik VM," presented by Ben Cheng and Bill Buzbee (Google's Android Team), *available at* http://developer.android.com/videos/index.html#v=Ls0tM-c4Vfo, at 50:53.)

47. I confirmed this performance impact by directing performance benchmark work conducted by Oracle Java engineer Bob Vandette in consultation with Oracle Java engineer Peter Kessler. The source code modifications were based on a number of discussions I had with Oracle Java engineers Bob Vandette and Peter Kessler, my review of the '205 patent, my review of Oracle's infringement contentions submitted to Google on April 1, 2011, and my understanding of the Android source code that satisfies the claims of the '205 patent.

48. The performance benchmark testing results show as much as 3.3 times execution speed improvement. This means applications can execute as much as 3.3 times faster.

49. It is my opinion that the performance of Android would be crippled without the benefit of using the '205 patent. Although Android did originally launch without a JIT, it would not continue to be a competitive platform in terms of execution speed and battery life without one.

50. I intend to rely on the summary and report submitted by Bob Vandette, submitted to Google with my report, to support my opinion.

51. ***'702 patent (Class File Redundancy Removal).*** In my opinion, the claimed invention of the '702 patent is essential to the performance of any Android device or Dalvik virtual machine execution environment. Google designed Android to take advantage of existing Java compilers by having source code written in the Java language first compiled to Java class files and then the class files translated to Google's .dex file format, rather than spend the substantial time and resources it would have taken to create a compiler that could compile directly from Java source code to the .dex file format (time and resources that I understand Google did not have). Given this Hobson's choice of using Java class files, in the absence of using the claimed invention, which reduces the size of class files by removing duplicate elements, the size of .dex files would be substantially larger. Larger application files would slow the speed of downloading an application to an Android device, reduce the ability to store multiple applications on an Android device, and slow the process of loading and executing applications on Android's Dalvik virtual machine.

52. I confirmed my qualitative analysis through directing performance benchmark work conducted by Oracle Java engineer Noel Poore. The source code modifications were based on a number of discussions I had with Noel Poore, my review of the '702 patent, my review of Oracle's infringement contentions submitted to Google on April 1, 2011, and my understanding of the Android source code that satisfies the claims of the '702 patent.

53. The performance benchmark testing results shows that a .dex file for an application is between 1.45 and 3.33 times smaller than it would be if duplicate constant removal were not performed. This means applications can be downloaded, stored, and executed in a

substantially smaller package with duplicates removed. It is my opinion that the performance of Android would be crippled without the benefit of using the '702 patent.

54. I intend to rely on the summary and report submitted by Noel Poore, submitted to Google with my report, to support my opinion.

55. ***'520 patent (Play Execution).*** In my opinion, the claimed invention of the '520 patent is important to the performance of any Android device or Dalvik virtual machine execution environment for applications that involve initialization of static arrays of primitive data types. In the absence of using the claimed invention, such application files in the .dex file format would be larger because they would contain additional initialization code, which means that such Android applications would take up more space and load more slowly.

56. I confirmed my qualitative analysis through directing performance benchmark work conducted by Oracle Java engineer Noel Poore. The source code modifications were based on a number of discussions I had with Noel Poore, my review of the '520 patent, my review of Oracle's infringement contentions submitted to Google on April 1, 2011, and my understanding of the Android source code that satisfies the claims of the '520 patent.

57. The performance benchmark testing compared the size in bytes of .dex files both with and without array initialization optimization. The testing showed that all .dex files were larger without the array initialization and that the cost for not using the array initialization optimization for larger .dex files is greater. This means that memory requirements and loading of applications would suffer as a result of not using the invention of the '520 patent. It is my opinion that the performance of Android would suffer as indicated in the performance analysis without the benefit of using the '520 patent.

58. I intend to rely on the summary and report submitted by Noel Poore, submitted to Google with my report, to support my opinion.

59. ***'720 patent (Copy-on-Write Process).*** In my opinion, because Google chose a one application per virtual machine instance architecture, Google has to practice the '720 patent to achieve acceptable performance. Had Google chosen a multiple process per virtual machine

instance architecture, then Google would *not* be able to rely on process separation to achieve security and would be boxed into relying more heavily on the technology patented in the '447 and '476 patents. Also, Google's use of the '720 patent permits the ability to launch more applications simultaneously, where memory consumption per application and virtual machine instance is reduced.

60. I confirmed my qualitative analysis through directing performance benchmark work conducted by Oracle Java engineers, Erez Landau and Seeon Birger. The source code modifications were based on a number of discussions I had with Erez Landau and Seeon Birger, my review of the '720 patent, my review of Oracle's infringement contentions submitted to Google on April 1, 2011, and my understanding of the Android source code that satisfies the claims of the '720 patent.

61. The performance benchmark testing shows that Android's use of the '720 patent results in as much as 40% memory savings for Android, memory savings of 2MB per each additional running application, and also a 0.10 second per application launch time savings. (In other words, the performance benchmark testing shows that disabling the functionality that Oracle accuses of infringing the '720 patent increases memory consumption by 70%.) These benchmark results become even more significant for consumers who run many small applications each of which requires a small amount of memory.

62. It is my opinion that the performance of Android would be crippled without the benefit of the '720 patent.

63. I intend to rely on the summary and report submitted by Erez Landau, submitted to Google with my report, to support my opinion.

64. Furthermore, Google described a benefit of practicing the '720 patent at http://developer.android.com/resources/articles/multitasking-android-way.html:

> "Application switching on a mobile device is extremely critical; we target significantly less than 1 second to launch a new application. This is especially important when the user is switching between a few applications, such as switching to look at a new

> SMS message while watching a video, and then returning to that video. A noticeable wait in such situations will quickly make users hate you."

65. The Android functionality that permits rapid application startup and switching is called "zygote." Zygote, as I explain below, infringes the '720 patent. As demonstrated in performance benchmark testing, Android's use of zygote reduces application startup time by at least 100ms, which is a big part of the "significantly less than 1 second" that Android's programmers needed to achieve. The description on the website of how applications are force-killed by Android when appropriate, and then restarted, underscores how limited memory is and how important zygote is to Android.

66. Google further describes a benefit of practicing the '720 patent in Google I/O 2008 Video entitled "Anatomy and Physiology of an Android" at slide 82, presented by Patrick Brady, *available at* http://developer.android.com/videos/index.html#v=G-36noTCaiA ("Android Presentation"); and corresponding Google I/O 2008 Video entitled "Anatomy and Physiology of an Android," presented by Patrick Brady, http://developer.android.com/videos/index.html#v=G-36noTCaiA ("Android Video").

(Android Presentation, Slide 82)

67. And in corresponding Android Video at 44:25:

> "The init process starts up a really neat process called zygote. And as its name implies, zygote is really just the beginning of all of the rest of the Android platform. And so zygote is a nascent, ah, VM process that initializes a Dalvik VM and preloads a lot of these libraries….It uses copy-on-write to maximize re-use and minimize footprint so that data structures are shared and it won't do a full copy unless some of those data structures are to be modified."

2. The Claimed Inventions are Critical to Achieving a Satisfactory Security Framework

68. ***'447 patent (Fine-Grained Security).*** I understand that Google states that it does not use the java.security framework in the applications that it has developed. I have not taken steps to verify this statement from Google. However, there are several reasons why Google chose to implement the java.security framework in Android and have and will benefit from including java.security in its platform, even if its contention is correct. Android's java.security framework implementation is covered by all of the asserted claims.

81. When I removed ProtectionDomain.java, the compilation and build of the Android system failed since many other classes import ProtectionDomain.java. Additional code rewriting would be required to remove this feature from the source code. I understand that Google engineer Dan Bornstein testified that this has been done for future versions of Android that have yet to be open-sourced and I have not had an opportunity to review. (7/22/2011 Bornstein Dep. 114:7-21.)

C. The Claimed Inventions of the Patents-in-Suit Form a Basis for Consumer Demand by Developers and End-Users

82. ***Virtual Machine and Handset Architecture Posed Special Performance and Security Challenges.*** It is well known that cellular phones are less expensive and more widely available worldwide than networked computers. In addition, smartphones that support web browsing and email occupy an increasing portion of the cellular market. (This is supported by graphs and statistics I cited in describing the rise of Android in my Opening Copyright Report, for example.) Therefore, as Google recognized, worldwide trends and customer demand present a striking challenge and opportunity for Google to make its products and services available to a dramatically increasing market:

> "More individuals are using devices other than personal computers to access the internet. If users of these devices do not widely adopt versions of our … operating systems developed for these devices, our business could be adversely affected.
>
> The number of people who access the internet through devices other than personal computers, including …smart phones [and] handheld computers...has increased dramatically in the past few years. The lower resolution, functionality, and memory associated with some alternative devices make the use of our products and services through such devices more difficult and the versions of our products and services developed for these devices may not be compelling to users, manufacturers, or distributors of alternative devices… ." (Google's 2008 10K and 2010 10Q Forms, *available at* http://investor.google.com/documents/2008_google_annual_report.html and http://investor.google.com/documents/20100930_google_10Q.html.)

83. Google's effort to have users widely adopt its operating systems for mobile devices led to technical challenges for the Android platform. Because the virtual execution

environment is central to its platform, Google realized it was essential to have a virtual machine that ran efficiently, even on a less powerful CPU, and required as little memory as possible. In addition, because mobile devices run on battery power, it is essential to require as little power as possible. As explained in a Google I/O 2008 Video entitled "Dalvik Virtual Machine Internals," presented by Dan Bornstein, *available at* http://developer.android.com/videos/index.html#v=ptjedOZEXPM ("Dalvik Presentation"),

> "So the virtual machine, again, is designed based on the constraints of the platform and you can see a few of the key ones. We're assuming, not a particularly powerful CPU, not very much RAM especially by say today's desktop standards. …
>
> So battery, you know, batteries are getting better as time goes on just like any technology. Unfortunately it doesn't look like it's keeping up with Moore's Law, so, you know, 18 months your, your CPUs are getting faster, your memory density is getting, you know, twice, twice as much, but your battery life isn't quite keeping up. So that really, that really is a, a key concern for any kind of mobile development."

84. The reference to Moore's Law suggests that battery technology is not improving at the same rate as the CPU and memory.

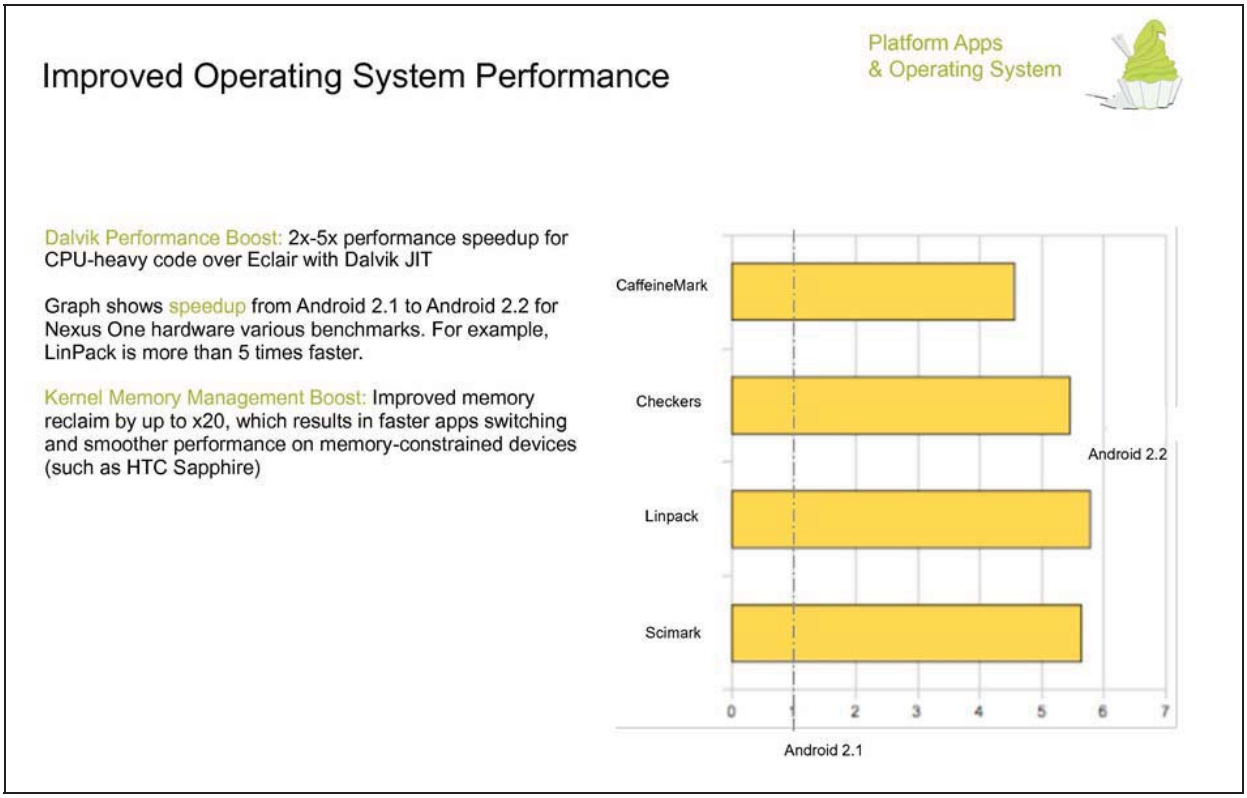
85. The particular requirements on the virtual machine are explained in a Google I/O 2010 Video entitled "A JIT Compiler for Android's Dalvik VM," presented by Ben Cheng and Bill Buzbee, *available at* http://developer.android.com/videos/index.html#v=Ls0tM-c4Vfo,

> "But what you have in effect there with interpretation is an extra stage of execution. So you have to pull up the bytecode, figure out what it is, then use host instructions to carry it out, and the CPU will pick up the host instructions and, and, and execute them. And that extra level of, of evaluation is what gives interpreters a bit of a bad name for being slow. I mean, there's some great reasons why you would want to use an interpreter, but the down side is that they're often a bit slower than native- natively compiled code."

Specifically, straightforward virtual machines use a bytecode interpreter. But, as explained in the quote by Android developer Bill Buzbee, this "extra stage of execution" slows down code execution. This problem with interpreters produces a need for execution improvements, such as

(Google Presentation titled "Highlights from Android OS Version 2.2 (Codenamed Froyo) (GOOGLE-03-00090810 at 813).)

91. Another performance chart from a Google Presentation titled "Highlights from Android OS Version 2.2 (Codenamed Froyo) (GOOGLE-03-00090810 at 817) similarly illustrates Google's concern with overall performance and the results of efforts to improve performance in Froyo:



As illustrated above, Google's own performance benchmarking shows consistent results as the performance benchmarking conducted at my request and under my direction.

92. Not only did Android engineers work to achieve acceptable performance of the core Android software, but the Google Android Compatibility Definition Document (*available at* http://source.android.com/compatibility/downloads.html) states that, "Compatible implementations must ensure not only that applications simply run correctly on the device, but that they do so with reasonable performance and overall good user experience." Therefore, Google understood that performance and security are key to satisfactory user experience.